IN THE CLAIMS

1. (Previously Presented) In a computer system, a method for managing services associated with a plurality of plug-in modules, the method comprising the steps of:

obtaining identities of a plurality of plug-in modules;

based on queries to the plurality of plug-in modules, retrieving a dependency list indicating respective plug-in services provided by, and required by, each plug-in module identified in the identities of a plurality of plug-in modules;

calculating a plug-in initiation order based upon the dependency list indicating respective plug-in services provided by, and required by, each plug-in module; and

initiating service operation of plug-in modules according to the plug-in initiation order, such that if a first plug-in module provides a service required by a second plug-in module, the first plug-in module is initiated such that the service provided by the first plug-in module is available to the second plug-in module when required by the second plug-in module.

2. (Previously Presented) The method of claim 1 wherein the step of obtaining identities of a plurality of plug-in modules includes the steps of:

receiving a list of services to be started within the computer system ;

determining, for each service in the list of services, a respective plug-in module definition that can provide that service; and

placing the identity of each plug-in module definition determined in the step of determining into the identities of the plurality of plug-in modules.

3. (Original) The method of claim 1 wherein the step of retrieving a dependency list indicating respective plug-in services provided by, and required by, each plug-in module comprises the steps of:

for each plug-in module identified in the identities of a plurality of plug-in modules, performing the steps of:

instantiating the plug-in module based upon a plug-in module definition associated with the identity of the plug-in module;

receiving a dependency response from the plug-in module, the dependency response indicating respective plug-in services provided by, and required by, the plug-in module; and

storing identities of the respective plug-in services provided by, and required by, the plug-in module as identified in the dependency response in the dependency list.

4. (Original) The method of claim 3 wherein the step of instantiating the plug-in module comprises the steps of:

obtaining plug-in initiation information corresponding to the plug-in module definition associated with the identity of the plug-in module;

instantiating the plug-in module based upon a plug-in module definition associated with the identity of the plug-in module; and

passing the plug-in initiation information to the plug-in module for use by the plug-in module.

5. (Original) The method of claim 3 wherein the step of instantiating the plug-in module comprises the step of:

querying a dependency interface associated with the plug-in module with a dependency query to obtain the dependency response from the plug-in module.

6. (Original) The method of claim 1 wherein the step of calculating a plug-in initiation order based upon the dependency list comprises the step of:

arranging a placement of each plug-in module identified in the dependency list within the plug-in initiation order such that plug-in modules not requiring services provided by other plug-in modules are placed earlier in the initiation order and such that plug-in modules requiring services provided by other plug-in modules are placed later in the initiation order.

7. (Original) The method of claim 6 wherein the step of arranging placement of each plug-in module identified in the dependency list within the plug-in initiation order comprises the steps of:

analyzing the dependency list indicating respective plug-in services provided by, and required by, each plug-in module to determine which plug-ins provide services relied upon by other plug-in modules; and

creating, as the plug-in initiation order, at least one plug-in module dependency tree based on the step of analyzing, the at least one plug-in module dependency tree containing a hierarchical arrangement of nodes associated with respective plug-in modules, the hierarchical arrangement indicating the plug-in initiation order of the plug-in modules respectively associated with the nodes in the dependency tree.

8. (Previously Presented) The method of claim 7 wherein initiating service operation of plug-in modules according to the plug-in initiation order comprises:

traversing the at least one plug-in module dependency tree according to the hierarchical arrangement of nodes and for each node encountered during the step of traversing, initiating service operation of the respective plug-in module associated with that node.

9. (Previously Presented) The method of claim 8 wherein the step of initiating service operation of plug-in modules includes:

forwarding, via a dependency available interface associated with a respective plug-in module, a list of initiated plug-in services of other plug-in modules that are currently available for use by the respective plug-in module.

10. (Original) The method of claim 1 wherein the step of initiating service operation of plug-in modules according to the plug-in initiation order comprises performing, for each respective plug-in module in the plug-in initiation order, the steps of:

determining, from a published list of services available by initiated plug-in modules, an identity of each initiated plug-in service required by the respective plug-in module;

forwarding to the respective plug-in module, via a dependency available interface associated with the respective plug-in module, the identity of each initiated plug-in service required by the respective plug-in module;

receiving a list of services initiated by the respective plug-in module; and

adding the list of services provided by the respective plug-in module to the published list of services.

11. (Original) The method of claim 1, wherein the step of initiating service operation of plug-in modules according to the plug-in initiation order operates such that if the second plug-in module requires a service provided by the first plug-in module, the second plug-in module is initiated such that the service provided by the first plug-in module is available to the second plug-in module when required.

12. (Previously Presented) The method of claim 1 wherein the first plug-in module is initiated via the step of initiating service operation of plug-in modules prior to initiation of the second plug-in module.

13. (Original) The method of claim 1 wherein the first plug-in module is initiated via the step of initiating operation of plug-in modules after initiation of the second plug-in module, and wherein the second plug-in module includes a wait-state operation causing the second plug-in module to wait to provide the service offered by the second plug-in module until initiation of the first plug-in module.

14. (Original) The method of claim 1 wherein the steps of obtaining, retrieving, calculating and initiating are performed by a multi-threaded plug-in manager and wherein the step of calculating a plug-in initiation order is performed by collectively operating a respective thread for each plug-in, each thread performing the step of initiating service operation of at least one plug-in module when all services required by that plug-in module are available.

15. (Previously Presented) A computer system comprising:

a memory;

a processor; and

an interconnection mechanism coupling the memory and the processor;

wherein the memory is encoded with a plug-in manager application that, when performed on the processor, produces a plug-in manager process that manages services associated with a plurality of plug-in modules encoded within the memory by performing the operation steps of:

obtaining identities of a plurality of plug-in modules in the memory;

based on queries to the plurality of plug-in modules, retrieving, into the memory, a dependency list indicating respective plug-in services provided by, and required by, each plug-in module identified in the identities of a plurality of plug-in modules;

calculating, in the memory, a plug-in initiation order based upon the dependency list indicating respective plug-in services provided by, and required by, each plug-in module; and

initiating service operation of plug-in modules on the processor according to the plug-in initiation order, such that if a first plug-in module provides a service required by a second plug-in module, the first plug-in module is initiated such that the service provided by the first plug-in module is available to the second plug-in module when required by the second plug-in module.

16. (Previously Presented) The computer system of claim 15 wherein when the plug-in manager process performs the step of obtaining identities of a plurality of plug-in modules, the plug-in manager process performs the steps of:

receiving a list of services to be started within the computer system;

determining, for each service in the list of services, a respective plug-in module definition that can provide that service; and

placing the identity of each plug-in module definition determined in the step of determining into the identities of the plurality of plug-in modules.

17. (Original) The computer system of claim 15 wherein when the plug-in manager process performs the step of retrieving a dependency list indicating respective plug-in services provided by, and required by, each plug-in module, the plug-in manager process performs the steps of:

for each plug-in module identified in the identities of a plurality of plug-in modules, performing the steps of:

instantiating the plug-in module in the memory based upon a plug-in module definition associated with the identity of the plug-in module;

receiving a dependency response from the plug-in module, the dependency response indicating respective plug-in services provided by, and required by, the plug-in module; and

storing, in the memory, identities of the respective plug-in services provided by, and required by, the plug-in module as identified in the dependency response in the dependency list.

18. (Original) The computer system of claim 17 wherein when the plug-in manager process performs the step of instantiating the plug-in module, the plug-in manager process performs the steps of:

obtaining, in the memory, plug-in initiation information corresponding to the plug-in module definition associated with the identity of the plug-in module;

instantiating the plug-in module in the memory based upon a plug-in module definition associated with the identity of the plug-in module; and

passing the plug-in initiation information to the plug-in module in the memory for use by the plug-in module.

19. (Original) The computer system of claim 17 wherein when the plug-in manager process performs the step of instantiating the plug-in module, the plug-in manager process performs the step of:

querying a dependency interface associated with the plug-in module with a dependency query to obtain the dependency response from the plug-in module.

20. (Original) The computer system of claim 15 wherein when the plug-in manager process performs the step of calculating a plug-in initiation order based upon the dependency list, the plug-in manager process performs the step of:

arranging a placement of each plug-in module identified in the dependency list within the plug-in initiation order such that plug-in modules not requiring services provided by other plug-in modules are placed earlier in the initiation order and such that plug-in modules requiring services provided by other plug-in modules are placed later in the initiation order.

21. (Original) The computer system of claim 20 wherein when the plug-in manager process performs the step of arranging placement of each plug-in module identified in the dependency list within the plug-in initiation order, the plug-in manager process performs the steps of:

analyzing the dependency list indicating respective plug-in services provided by, and required by, each plug-in module to determine which plug-ins provide services relied upon by other plug-in modules; and

creating in the memory, as the plug-in initiation order, at least one plug-in module dependency tree based on the step of analyzing, the at least one plug-in module dependency tree containing a hierarchical arrangement of nodes associated with respective plug-in modules, the hierarchical arrangement indicating the plug-in initiation order of the plug-in modules respectively associated with the nodes in the dependency tree.

22. (Original) The computer system of claim 21 wherein when the plug-in manager process performs the step of initiating service operation of plug-in modules according to the plug-in initiation order, the plug-in manager process performs the step of:

traversing the at least one plug-in module dependency tree in the memory according to the hierarchical arrangement of nodes and for each node encountered during the step of traversing, initiating service operation of the respective plug-in module associated with that node.

23. (Original) The computer system of claim 22 wherein when the plug-in manager process performs the step of initiating service operation of the respective plug-in module associated with that node, the plug-in manager process performs the step of:

forwarding, via a dependency available interface associated with the respective plug-in module, a list of initiated plug-in services of other plug-in modules that are currently available for use by the respective plug-in module.

24. (Original) The computer system of claim 15 wherein when the plug-in manager process performs the step of initiating service operation of plug-in modules according to the plug-in initiation order the plug-in manager process

performs, for each respective plug-in module in the plug-in initiation order, the steps of:

determining, from a published list of services available by initiated plug-in modules, an identity of each initiated plug-in service required by the respective plug-in module;

forwarding to the respective plug-in module, via a dependency available interface associated with the respective plug-in module, the identity of each initiated plug-in service required by the respective plug-in module;

receiving a list of services initiated by the respective plug-in module; and

adding the list services provided by the respective plug-in module to the published list of services.

25. (Original) The computer system of claim 15, wherein the step of initiating service operation of plug-in modules according to the plug-in initiation order operates in the plug-in manager process such that if the second plug-in module requires a service provided by the first plug-in module, the second plug-in module is initiated such that the service provided by the first plug-in module is available to the second plug-in module when required.

26. (Previously Presented) The computer system of claim 15 wherein the plug-in manager initiates the first plug-in module via the step of initiating service operation of plug-in modules prior to initiation of the second plug-in module.

27. (Original) The computer system of claim 15 wherein the plug-in manager process initiates the first plug-in module via the step of initiating operation of plug-in modules after initiation of the second plug-in module, and wherein the second plug-in module includes a wait-state operation causing the second plug-in module to wait to provide the service offered by the second plug-in module until initiation of the first plug-in module.

28. (Original) The computer system of claim 15 wherein the steps of obtaining, retrieving, calculating and initiating are performed by a multi-threaded plug-in manager and wherein the step of calculating a plug-in initiation order is performed by collectively operating a respective thread for each plug-in, each thread performing the step of initiating service operation of at least one plug-in module when all services required by that plug-in module are available.

29. (Previously Presented) A computer program product having a computer-readable medium including computer program logic encoded thereon, that when executed on a computer system having a coupling of a memory and a processor, provides a plug-in manager process for managing plug-in services by causing the processor to perform the operations of:

obtaining identities of a plurality of plug-in modules in the memory;

retrieving, into the memory, a dependency list indicating respective plug-in services provided by, and required by, each plug-in module identified in the identities of a plurality of plug-in modules;

calculating, in the memory, a plug-in initiation order based upon the dependency list indicating respective plug-in services provided by, and required by, each plug-in module;

initiating service operation of plug-in modules on the processor according to the plug-in initiation order, such that if a first plug-in module provides a service required by a second plug-in module, the first plug-in module is initiated such that the service provided by the first plug-in module is available to the second plug-in module when required by the second plug-in module; and

querying a dependency interface associated with the plug-in module with a dependency query to obtain a dependency response from the plug-in module, the dependency response indicating respective plug-in services provided by the plug-in module.

30. (Previously Presented) A computer system comprising:

a memory;

a processor; and

an interconnection mechanism coupling the memory and the processor;

wherein the memory is encoded with a plug-in manager application that, when performed on the processor, produces a plug-in manager process that manages services associated with a plurality of plug-in modules encoded within the memory by operating on the computer system and causing the computer system to provide:

means for obtaining identities of a plurality of plug-in modules in the memory;

means for retrieving, into the memory, a dependency list indicating respective plug-in services provided by, and required by, each plug-in module identified in the identities of a plurality of plug-in modules;

means for calculating, in the memory, a plug-in initiation order based upon the dependency list indicating respective plug-in services provided by, and required by, each plug-in module;

means for initiating service operation of plug-in modules on the processor according to the plug-in initiation order, such that if a first plug-in module provides a service required by a second plug-in module, the first plug-in module is initiated such that the service provided by the first plug-in module is available to the second plug-in module when required by the second plug-in module; and

wherein the first plug-in module is initiated via the step of initiating operation of plug-in modules after initiation of the second plug-in module, and wherein the second plug-in module includes a wait-state operation causing the second plug-in module to wait to provide the service offered by the second plug-in module until initiation of the first plug-in module.

31. (Previously Presented) A computer program product as in claim 29, wherein the processor further performs operations of:

determining a list of plug-in services required by a software application; and

querying a set of plug-in modules to identify services provided by the set of plug-in modules.

32. (Previously Presented) A computer program product as in claim 31, wherein the processor further performs operations of:

in response to querying the set of plug-in modules, identifying plug-in modules not identified by the software application as being necessary but which are identified by the set of plug-in modules as being necessary to carry out execution of an operation on behalf of the software application.

33. (Previously Presented) A computer program product as in claim 32, wherein the processor further performs operations of:

initiating service operation of plug-in modules on the processor according to an order other than the plug-in initiation order, such that if a third plug-in module provides a service required by a fourth plug-in module, the third plug-in module being initiated after initiation of the fourth plug-in module, the third plug-in module initiating a wait state operation causing the third plug-in module to wait to provide the service offered by the third plug-in module until instantiation of the fourth plug-in module.

34. (Previously Presented) A computer program product as in claim 32, wherein the processor initiates execution of the first plug-in module before execution of the second plug-in module, the first plug-in module initiating a wait state operation resulting in signaling to the second plug-in module,

the signaling indicating that a respective service of the first plug-in module is not yet available to the second plug-in module.

35. (Previously Presented) A computer program product as in claim 34, wherein the processor further performs operations of:

maintaining a list of services for a set of plug-in modules currently able to provide respective services; and

publishing the list of services for the software application to identify instantiated plug-ins currently providing the respective services.